Proyecto Final de **Estructura de datos**

Titulo del Proyecto : **Administrador de memoria**

Lenguaje C, listas doblemente ligadas, aritmetica de apuntadores, arreglos

Objetivo: Gestionar la memoria usando los algoritmos **mejor ajuste o primer ajuste:**

La representación de la memoria se puede implementar mediante una lista doblemente enlazada con el inicio, el límite del bloque y el tipo hueco (H) o proceso (P). El tipo, protege de ser sobreescritos a aquellos bloques que ya fueron asignados. La función que asigna memoria debe regresar un entero (el inicio), el cual servirá como referencia para el usuario al liberar bloques de la misma.

La representación del estado de la memoria se puede considerar de la manera siguiente.

En la figura los espacios en gris (rayados) son la representación de los huecos, mientras que los demás

bloques denotados por una letra son los procesos cargados por el usuario.

Para la asignación de memoria use uno de los 2 algoritmos siguientes: el mejor ajuste y el primer ajuste.

**El mejor ajuste:** Recorre toda la lista encontrando los agujeros. El algoritmo elegirá el espacio que

mejor se ajuste al requerido por el proceso. Una de las desventajas más grandes de este algoritmo es la

fragmentación que produce a la memoria, dejando espacios muy pequeños que pocos procesos podrían

utilizar en futuras asignaciones.

**El primer ajuste:** Este algoritmo igualmente recorre la lista, pero a diferencia del mejor ajuste, este

algoritmo asigna el primer espacio de memoria disponible que se encuentre. En primera instancia se

puede pensar que en ciertos casos se desperdicia memoria al asignar grandes huecos a procesos

pequeños. Por otro lado la memoria no se fragmenta de igual manera que en el mejor ajuste. Esto no se

puede evitar, pero podría tardar más tiempo, dependiendo de las asignaciones.

La asignación de memoria requiere de casos completamente diferentes, los cuales se multiplican debido

a la posibilidad de que los apuntadores que siguen a los nodos ubicados en posiciones como el inicio o

el final de la memoria, suelen apuntar a NULL (nulo), por lo que si se intenta acceder a una parte de la

estructura como la variable tipo, dentro de esta clase de apuntadores se tendrá como resultado una

violación de segmento.

Al momento de realizar las comprobaciones se deben colocar en un orden específico de manera que no

se llegue al caso anterior. Primero se deberá verificar que ninguno de los dos apuntadores es nulo,

posteriormente si uno de los dos lo és y al final se podrán verificar los tipos de los nodos anteriores y

siguientes.

Se deberán desligar los nodos que se van a liberar, de la lista principal. Cuando libere memoria considere el caso en que los nodos son del tipo agujero y se deben unir.

Una manera multiplataforma de representar gráficamente el mapa de memoria podría ser con caracteres

ASCII y la librería ncurses:

El administrador debe aceptar al menos dos ordenes: asignar y liberar la memoria:

Al mismo tiempo comenzará a desplegar los espacios con el respectivo inicio del bloque así como el

tamaño de la misma. A continuación de muestra el despliegue de las asignaciones de la memoria:

Documentación a entregar

-Comentarios en código fuente.

-**Manual de usuario** impreso mínimo 2 paginas máximo 5 paginas

incluir las pantallas necesarias

-M**anual técnico** impreso mínimo 3 paginas máximo 10 paginas.

-Diagrama de clases.

-Dar una explicación de alto nivel de como se realizo el proyecto e incluir

los fragmentos de código mas importantes para complementar dicha

explicación

**Entregar**

Un **CD** o **miniCD** o **DVD** o **miniDVD** o **USB**

-codigo fuente y ejecutable

-archivos: txt y todos aquellos recursos necesarios para ejecutar el proyecto

-Manuales

-Portada con la siguiente información:

nombre alumno, grupo donde tomo el curso, nombre proyecto, fecha y Paradigmas

de Programacion (materia).